# Software Paradigms SS 2015 2

| Nachname | Vorname | Matrikelnummer |
|---|---|---|
| LORENZ | Peter | |
| ZIKO | Haris | |

## Exercise 1

```
begin
z := 1
foreach i in [1,2,3] do
  z := mul(z, i)
end
```

Def.: formale Semantik für das Statement ($v \in IVS, t_1 \in T_L, a \in A$)

1. Assignements: $I_A(\omega$, v $\underline{:=}$ t) = w' $_v$ $\omega$ und $w'(v)$=$I_T(\omega, t)$.
2. Schleifen: $I_A(\omega$, $\underline{\text{foreach}}$ $t_i$ $\underline{\text{do}}$ a) = $I_A(I_A(\omega, a)$, $\underline{\text{foreach v do a}}$) wenn $I_T(\omega, t)$ mit t $\neq \varepsilon$.
3. mul: definiert im Skriptum v. D. Gruss (S.29)

$I_A(\omega$, $\underline{\text{begin z := 1; foreach v in [1,2,3] do z := mul(z,v)}}$
= $I_A(I_A(\omega$, $\underline{\text{z := 1}}$), $\underline{\text{foreach v in [1,2,3] do}}$ z:= mul(z,v))

NE: $\omega^1 \sim_{\underline{x}} \omega$ und berechnen $\omega^1$ ($\underline{\text{z}}$) = $I_T$=($\omega$,$\underline{1}$) = 1

= $I_A$ ($\omega^1$, $\underline{\text{foreach v do mul(z,v))}}$
= mul($\omega$ $^1$(underlinez), $\omega$ $^1$(underlinev)) = mul(1,1) = 1
NR: $I_T$ ($\omega^1$, $\underline{1}$) und $\underline{1} \neq \varepsilon$
= $I_A(I_A(\omega^1$, $\underline{\text{z := mul(z,v)}}$), foreach v do mul(z,v))

NE: $\omega^2 \sim_{\underline{z}} \omega$ und berechnen $\omega^1$ ($\underline{\text{z}}$) = $I_T$=($\omega$,$\underline{2}$) = 2
= mul($\omega$ $^2$(underlinez), $\omega$ $^2$(underlinev)) = mul(1,2) = 2
= $I_A(I_A(\omega^2$, $\underline{\text{z := mul(z,v)}}$), $\underline{\text{foreach v do mul(z,v))}}$

NE: $\omega^3 \sim_{\underline{z}} \omega$ und berechnen $\omega^3$ ($\underline{\text{z}}$) = $I_T$=($\omega$,$\underline{3}$) = 3
= mul($\omega$ $^3$(underlinez), $\omega$ $^3$(underlinev)) = mul(2,3) = 6
= $I_A(I_A(\omega^3$, $\underline{\text{z := mul(z,v)}}$), $\underline{\text{foreach v do mul(z,v))}}$

NE: $\omega^4 \sim_{\underline{z}} \omega$ und berechnen $\omega^3$ ($\underline{\text{z}}$) = $I_T$=($\omega$, $\underline{\varepsilon}$) = $\varepsilon$

Finished!

## Exercise 2

Encoding for elements in A:
prove, that this is true:
$\pi$((n,d)) = $build(n, build(d, []))$ = $[n, d]$

## Exercise 3

Encode for $f_1$:
$\pi$[insert](s, i) = if eq?(s, []) then build(i, []) else
if eq?(first(s), i) then s else build(first(s), insert(rest(s), i))
Dies ist eine rekursive Lösung, die als erstes Abfragt ob eine leere Menge gegeben ist, andererseits wird rekursiv überprüft, ob die Zahl i in der Menge mitenthalten ist. Wenn die Zahl nicht in der Liste ist, dann wir sie am Ende der List dazugehängt. Ansonsten wird die normale Liste zurückretuniert.

$\pi$[insert](s, i) = if eq?(s, []) then [] else
if eq?(first(s), i) then rest(s) else build(first(s), insert(rest(s), i))
Dies ist eine rekursive Lösung, die als erstes Abfragt ob eine leere Menge gegeben ist, andererseits wird rekursiv überprüft, ob die Zahl i in der Menge mitenthalten ist. Wenn die Zahl in der Liste enthalten ist, dann wird sie von der Liste entfernt. Ansonsten wird die normale Liste zurückretuniert.

$\pi$[isEmpty?](s) = if eq?(s, []) then T else F
Hier wird überprüft, ob die Liste leer ist oder nicht. Wenn sie leer ist, dann wird True zurückgegeben, ansonsten False.
$\pi$[isElement?](s, i) = if eq?(s, []) then F else if eq?(first(s), i) then T else isElement?(rest(s), i)
$\pi$((emptyS)) = if eq?(emptyS, []) then [] else emptyS

## Exercise 4

1st: $\underline{(\forall x)(\exists y)\ \text{eq?(mult(x,y),x)}}$

$I_{\mathcal{P}}(\omega, \underline{(\forall x)(\exists y)\ \text{eq?(mult(x,y),x)}})$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : I_{\mathcal{P}}(\omega', \underline{(\exists y)\ \text{eq?(mult(x,y),x)}})$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : \exists\omega'', \omega' \sim_{\underline{y}} \omega'' : I_{\mathcal{P}}(\omega'', \underline{\text{eq?(mult(x,y),x)}})$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : \exists\omega'', \omega' \sim_{\underline{y}} \omega'' :\text{eq?}(I_{\mathcal{P}}(\omega'', \underline{\text{(mult(x,y),x)}}))$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : \exists\omega'', \omega' \sim_{\underline{y}} \omega'' : \text{eq?}(I_{\mathcal{P}}(\omega'', \underline{\text{(mult(x,y))}}), I_{\mathcal{P}}(\omega'', \underline{\text{x}}))$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : \exists\omega'', \omega' \sim_{\underline{y}} \omega'' : \text{eq?(mult(}I_{\mathcal{P}}(\omega'',\underline{\text{x}}),I_{\mathcal{P}}(\omega'',\underline{\text{y}})),\ I_{\mathcal{P}}(\omega'', \underline{\text{x}}))$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : \exists\omega'', \omega' \sim_{\underline{y}} \omega'' : \text{eq?(mult(}\omega''(\underline{\text{x}}),\omega''(\underline{\text{y}})),\omega''(\underline{\text{x}}))$

When we say, that $\omega''(\underline{\text{x}})$ = x and $\omega''(\underline{\text{y}})$ = y, then we can say that $x \cdot y = x$ only when y is 1.
We can say for $(\forall x)(\exists y) x \cdot y = x$ only true, if and only if $y = 1$.
Therefore we can say that:
$\Longleftrightarrow T$
is valid

2nd: $\underline{(\forall x)\text{eq?(build(x,nil),nil)}\vee\text{eq?(1,x)}}$

$\Longleftrightarrow\ I_{\mathcal{P}}(\omega, \underline{(\forall x)\text{eq?(build(x,nil),nil)}\vee\text{eq?(1,x)}})$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : I_{\mathcal{P}}(\omega', \underline{\text{eq?(build(x,nil),nil)}\vee\text{eq?(1,x)}})$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : I_{\mathcal{P}}(\omega',\underline{\text{eq?(build(x,nil),nil)}}))\vee I_{\mathcal{P}}(\omega',\underline{\text{eq?(1,x)}})$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : I_{\mathcal{P}}(\omega',\underline{\text{eq?(build(x,nil),nil)}}))\vee \text{eq?}(I_{\mathcal{P}}(\omega',\underline{1}),I_{\mathcal{P}}(\omega',\underline{\text{x}}))$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : \text{eq?}(I_{\mathcal{P}}(\omega',\underline{\text{build(x,nil)}}),I_{\mathcal{P}}(\omega',\underline{\text{nil}}))\vee \text{eq?}(I_{\mathcal{P}}(\omega',\underline{1}),\omega'(\underline{\text{x}}))$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : \text{eq?(build(}I_{\mathcal{P}}(\omega',\underline{\text{x}}),I_{\mathcal{P}}(\omega',\underline{\text{nil}})),I_{\mathcal{P}}(\omega',\underline{\text{nil}}))\vee \text{eq?}(\omega'(\underline{1}),\omega'(\underline{\text{x}}))$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : \text{eq?(build(}I_{\mathcal{P}}(\omega',\underline{\text{x}}),I_{\mathcal{P}}(\omega',\underline{\text{nil}})),\omega'(\underline{\text{nil}}))\vee \text{eq?}(\omega'(\underline{1}),\omega'(\underline{\text{x}}))$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : \text{eq?(build(}I_{\mathcal{P}}(\omega',\underline{\text{x}}),\omega'(\underline{\text{nil}})),\omega'(\underline{\text{nil}}))\vee \text{eq?}(\omega'(\underline{1}),\omega'(\underline{\text{x}}))$
$\Longleftrightarrow\ \forall\omega', \omega \sim_{\underline{x}} \omega' : \text{eq?(build(}\omega'(\underline{\text{x}}),\omega'(\underline{\text{nil}})),\omega'(\underline{\text{nil}}))\vee \text{eq?}(\omega'(\underline{1}),\omega'(\underline{\text{x}}))$

We can now say that $\omega'(\underline{x}) = x$ and $\omega'(nil) = []$, then we get

$\iff \forall \omega', \omega \sim_{\underline{x}} \omega'$ : eq?(build(x,[]),[])$\vee$eq?(1,x)

When we write it more mathematically, it will look like

$\forall x : ([x] = []) \vee 1 = x$

For $\forall$ we only need one contra example, e.g. $x = 2$ we can see that this make the whole statement False therefore we can say:

$\iff F$

3rd:$(\forall x)$($\neg$ eq?(x,y) $\rightarrow$ gt?(x,y)) $\ldots$ Datentyp $N$
$I_{\mathcal{P}}(\omega,$ $\underline{(\forall x)(\neg \text{ eq?(x,y)} \rightarrow \text{gt?(x,y)})}$)

$\iff \forall \omega', \omega \sim_{\underline{x}} \omega'$ : $I_{\mathcal{P}}(\omega'$,$\neg$ eq?(x,y) $\rightarrow$ gt?(x,y)))
$\iff \forall \omega', \omega \sim_{\underline{x}} \omega'$ : $I_{\mathcal{P}}(\omega'$, $\neg$ eq?(x,y)) $\rightarrow$ $I_{\mathcal{P}}$gt?(x,y))
$\iff \forall \omega', \omega \sim_{\underline{x}} \omega'$ : $\neg I_{\mathcal{P}}(\omega'$, eq?(x,y)) $\rightarrow$ $I_{\mathcal{P}}(\omega'$, gt?(x,y))
$\iff \forall \omega', \omega \sim_{\underline{x}} \omega'$ : $\neg$ eq?($I_{\mathcal{P}}$(x,y)) $\rightarrow$ gt?($I_{\mathcal{P}}$(x,y))
$\iff \forall \omega', \omega \sim_{\underline{x}} \omega'$ : $\neg$ eq?(x,y) $\rightarrow$ gt?(x,y)